Accelerated NVMe-oF target and vhost via SPDK

Ziye Yang, Changpeng Liu, Paul E Luse
Intel
Agenda

- What is SPDK
- Accelerated NVMe-oF target/host
- Accelerated vhost target
- Summary
Agenda

- What is SPDK
- Accelerated NVMe-oF target/host
- Accelerated vhost target
- Summary
The Problem: Software is becoming the bottleneck

Latency
- HDD: >2ms
- SATA NAND SSD: <100µs
- NVMe* NAND SSD: <100µs
- Intel® Optane™ SSD: >400,000 IO/s

I/O performance
- HDD: <500 IO/s
- SATA NAND SSD: >25,000 IO/s
- NVMe* NAND SSD: >400,000 IO/s
- Intel® Optane™ SSD: <100µs

The Opportunity: Use Intel software ingredients to unlock the potential of new media
Storage Performance Development Kit

Intel® Platform Storage Reference Architecture

- Optimized for Intel platform characteristics
- Open source building blocks (BSD licensed)
- Available via github.com/spdk or spdk.io

Scalable and Efficient Software Ingredients

- User space, lockless, polled-mode components
- Up to millions of IOPS per core
- Designed for Intel Optane™ technology latencies
Benefit of using SPDK

- Up to **10X MORE** IOPS/core for NVMe-oF* vs. Linux kernel
- Up to **8X MORE** IOPS/core for NVMe vs. Linux kernel
- Up to **350% BETTER** Tail Latency for RocksDB workloads
- Faster TTM/ less resources than developing components from scratch
- Provides future proofing as NVM technologies increase in performance

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to [http://www.intel.com/performance](http://www.intel.com/performance).
Architecture

Storage Protocols
- iSCSI Target
- vhost-scsi Target
- NVMe-oF* Target
- vhost-blk Target
- Object
- SCSI
- NVMe

Storage Services
- Block Device Abstraction (BDEV)
- 3rd Party
- Blob bdev
- NVMe
- Linux Async IO
- Ceph RBD
- BlobFS
- Blobstore

Drivers
- NVMe Devices
  - NVMe-oF* Initiator
  - NVMe* PCIe Driver
  - Intel® QuickData Technology Driver

Integration
- Released Q3'17 Pathfinding
- RocksDB
- Ceph

Core
- Application Framework
SDPK current development status

- Fully realizing new media performance requires software optimizations
- SPDK positioned to enable developers to realize this performance
- SPDK available today via http://spdk.io
- Help us build SPDK as an open source community!
Agenda

- What is SPDK
- Accelerated NVMe-oF target/host
- Accelerated vhost target
- Summary
SPDK NVMe-oF Components

- **NVMe over Fabrics Target**
  - Released July 2016 (with spec)
  - **17.03 Hardening:**
    Intel test infrastructure
    Discovery simplification
    Correctness & kernel interop
  - **17.03 Performance:**
    Read latency improvement
    Scalability validation (up to 150Gbps)
    Event Framework enhancements

- **NVMe over Fabrics Host (Initiator)**
  - New component added in 16.12
  - Performance improvements in 17.03:
    Eliminate copy: now true zero-copy SGL (single SGL element)
NVMe-oF Target Throughput Performance comparison

### SPDK vs. Kernel NVMe-oF I/O Efficiency

<table>
<thead>
<tr>
<th>IOPS</th>
<th>Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>30</td>
</tr>
</tbody>
</table>

#### NVMe* over Fabrics Target Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Realized Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utilizes NVM Express* (NVMe) Polled Mode Driver</td>
<td>Reduced overhead per NVMe I/O</td>
</tr>
<tr>
<td>RDMA Queue Pair Polling</td>
<td>No interrupt overhead</td>
</tr>
<tr>
<td>Connections pinned to CPU cores</td>
<td>No synchronization overhead</td>
</tr>
</tbody>
</table>

SPDK reduces NVMe over Fabrics software overhead up to 10x!

System Configuration: Target system: Supermicro SYS-2028U-TN24R4T+, 2x Intel® Xeon® E5-2699v4 (HT off), Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 8x 8GB DDR4 2133 MT/s, 1 DIMM per channel, 12x Intel® P3700 NVMe SSD (800GB) per socket, -1H0 FW; Network: Mellanox® ConnectX-4 LX 2x25Gb RDMA, direct connection between initiators and target; Initiator OS: CentOS® Linux® 7.2, Linux kernel 4.10.0, Target OS (SPDK): Fedora 25, Linux kernel 4.9.11, Target OS (Linux kernel): Fedora 25, Linux kernel 4.9.11. Performance as measured by: fio, 4KB Random Read I/O, 2 RDMA QP per remote SSD, Numjobs=4 per SSD, Queue Depth: 32/job. SPDK commit ID: 4163626c5c.
NVMe-oF Target Latency

Comparison to Linux Kernel 4.10.1 (hybrid polling enabled)

SPDK Target Eliminates ~3-5 usec of protocol + driver latency per I/O

Avg. Read I/O Round Trip Time
Kernel vs. SPDK NVMe-oF Target
Coldstream, Perf, qd=1

Avg. Write I/O Round Trip Time
Kernel vs. SPDK NVMe-oF Target
Coldstream, Perf, qd=1

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW=14.16.1020, mlx5_core=3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/O 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depth: 1/NVMe-oF subsystem. numjobs 1, 300 sec runtime, direct=1, norandommap=1, FIO-2.12, SPDK commit # 42eade49
NVMe-oF Host (Initiator) Latency

Comparison to Linux Kernel 4.10.1 (hybrid polling enabled)

Fabric + Software overhead: 33%-46% reduction per transaction using SPDK

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW= 14.16.1020, mlx5_core= 3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/O 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depth: 1/NVMe-oF subsystem. numjobs 1, 300 sec runtime, direct=1, norandommap=1, FIO-2.12, SPDK commit.
SPDK Host + Target vs. Kernel Host + Target

Avg. I/O Round Trip Time

Kernel vs. SPDK NVMe-oF Stacks

Coldstream, Perf, qd=1

SPDK reduces Optane NVMe-oF latency by 44%, write latency by 32%!

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW= 14.16.1020, mlx5_core= 3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/O 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depth: 1/NVMe-oF subsystem. numjobs 1, 300 sec runtime, direct=1, norandommap=1, FIO-2.12, SPDK commit # 42eade49
NVMe-oF IO Latency Model, 4KB Random Read (Intel Optane SSD DC P4800X)

SPDK Target + Kernel NVMe-oF Initiator

IO Read from Kernel NVMe-oF Initiator initiator

Start: 0 usec

20 usec

nvmf_READ I/O start

6 usec

nvmf_READ I/O complete

13.15 usec

SPDK NVMf Target

spdk_lib_read_start

6.05 usec

spdk_lib_read_complete

13.05 usec

SPDK NVMe lib

Optane Controller

Latency Distribution

- 20 usec round trip time measured from NVMe-oF initiator
- Out of 20 usec, ~7 usec spent in NVMe controller
- 12-13 usec measured time in the fabric and kernel NVMe-oF initiator
- SPDK NVMf target adds just 100-200 nsec to fabric overhead

Disclaimer: Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW=14.16.1020, mlx5_core=3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/O 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depth: 1/NVMe-oF subsystem. numjobs 1, 300 sec runtime, direct=1, norandommap=1, FIO 2.12, SPDK commit # 42eade49
NVMe-oF IO Latency Model, 4KB Random Read (Intel Optane SSD DC P4800X)

**SPDK Target + SPDK NVMe-oF Initiator**

- IO Read from SPDK NVMe-oF Initiator initiator
- Start: 0 usec
- 14 usec

- nvmf_read I/O start
  - 2 usec
- nvmf_read I/O complete
  - 9.15 usec

- SPDK NVMe Target
  - spdk_lib_read_start
  - 2.05 usec
  - spdk_lib_read_complete
  - 9.05 usec

- SPDK NVMe lib

- Optane Controller

---

**Latency Distribution**

- Fabric Depart, 4.85 usec
- Device, 7 usec
- SPDK Submit, 0.05

- Fabric Arrival, 2 usec

- Start
- Fabric Arrival
- SPDK Submit
- Device
- SPDK Complete
- Fabric Depart

- latency: Time in usec
- latency distribution

- 14 usec round trip time measured from NVMf client
- Out of 14 usec, ~7 usec spent in NVMe controller
- 7 usec measured time in the fabric and SPDK NVMe-oF initiator
- SPDK NVMe target adds just 100–200 nsec to fabric overhead

---

Disclaimer: Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s; Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW=14.16.1020, mlx5_core=3.0-1 driver, 1 ColdStream, connected to socket 0; 4KB Random Read (IO 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read (IO, Queue Depth: 1/NVMe-oF subsystems: numjobs 1, 300 sec runtime, direct=1, norandommap=1, PIO=2-12, SPDK commit # 42eade49

---

2017 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.
Agenda

- What is SPDK
- Accelerated NVMe-oF target/host
- Accelerated vhost target
- Summary
VHOST Introduction

- Separate process for I/O processing
- vhost protocol for communicating guest VM parameters
  - memory
  - number of virtqueues
  - virtqueue locations
SPDK VHOST Target

- Support virtio-scsi and virtio-blk in Guest
- QEMU sets up vhost target via UNIX domain socket memory
- Guest VM submits I/O directly to vhost target via virtqueues in shared memory, no Qemu intervention
- Completion interrupt sent using eventfd which does require system call and guest VM exit
- QEMU can pre-allocate huge pages for guest VM to enable direct DMA by vhost target
Comparison with Exist Solution

QEMU VIRTIO SCSI Target

VHOST Kernel Target

VHOST Userspace Target

QEMU
Guest VM
Guest Kernel
VIRTIO_SCSI
VIRTIO_SCSI_PCI
Host Kernel
NVME_MOD

QEMU
Guest VM
Guest Kernel
VIRTIO_SCSI
VHOST_SCSI_PCI
IOCTL
Host Kernel
VHOST
LIO
NVME_MOD

QEMU
Guest VM
Guest Kernel
VIRTIO_SCSI
VHOST_USER_SCSI_PCI
SPDK VHOST
VHOST_USER
SCSI
PMD_NVME

Host Kernel
NVME_MOD

VHOST
Userspace
Target

2017 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.
Vhost Benchmarks

configuration: 44x Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz (HT off); Cores per socket: 22; 8x Samsung 8GB DDR4 @2400 12x Intel SSD DC P3700 Series 1,5T @ FW 8DV101H0 DPDK: 17.02; Host Dist/Kernel: Fedora 25/kernel 4.8.15-300; Guest Dist/Kernel: Ubuntu 16.04/kernel 4.4.0-59-generic, mq enabled; Fio ver: fio-2.2.10; Fio workload: blocksize=4k, iodepth=512, iodepth_batch=128, iodepth_low=256, ioengine=libaio, size=10G, ramp_time=10, group_reporting, thread, numjobs=1, direct=1, rw=randread
Agenda

- What is SPDK
- Accelerated NVMe-oF target/host
- Accelerated vhost target
- Summary
In this presentation, we introduce the two accelerated apps built from SPDK and analyze the performance:

- Accelerated NVMe-oF target/host
- Accelerated vhost-scsi target

SPDK proves to be useful to accelerate storage applications equipped with NVMe based devices.

Call for action:
- Welcome to use SPDK and contribute into SPDK community